OPTIMIZATION ASPECTS OF DISTRIBUTED APPLICATIONS

Mihai Doinea[1]

Abstract

The paper presents a synthesis of the research in the area of distributed applications and security optimization. An aggregated indicator is presented and its characteristics discussed in order to use it as a point of reference for the assessment of the distributed applications quality. A set of quality metrics are presented and a conclusion is drawn upon the quality level of a distributed application. Using the indicator presented the security optimization effects in distributed applications are revealed. The optimization is presented as an automated process for a security control in a distributed application.

Keywords: distributed applications, optimization process, security, metrics.

1. Aggregated indicator for quality analysis

For analyzing the optimization effects for a distributed application the following aggregated indicator is used expressed as an indicator of type fraction, indicator presented in [IICM10]:

$$S_{x,y} = \frac{\min(x,y)}{\max(x,y)} \; with \; x > 0, y > 0$$

The variables x and y belong to R^*_+. When x < y, it results that $S_{x,y} = \frac{x}{y}$. For $x > y$, the value of the indicator is $S_{x,y} = \frac{y}{x}$ and, for $x = y$, $S_{x,y} = 1$. The value of $S_{x,y}$ tends to zero when a number is small and the other has a large value.

The presented indicator is analyzed with regards to its properties: sensitivity, non-compensatory, non-catastrophic and representativeness.

The **sensitivity** of an indicator represents the fact that, at small or large variations of x and y factors, same variations are reflected to the $S_{x,y}$ indicator.

For this analyse, a starting point is considered for the x and y factors. Multiple values of x and y are taken into consideration, simulating the variations among the exogenous factors. The results of the new values of the $S_{x,y}$ indicator are calculated and compared in order to accept or not the property of sensitivity for $S_{x,y}$ indicator.

The situation when x < y, $S_{x,y} = \frac{x}{y}$.

By modifying x input with a value of δ, $\delta > 0$ and $x + \delta < y$, the new indicator.

$$S_{x+\delta,y} = \frac{min(x+\delta, y)}{max(x+\delta, y)} = \frac{x+\delta}{y} = \frac{x}{y} + \frac{\delta}{y} = S_{x,y} + \frac{\delta}{y}$$

[1] Department of Informatics and Economic Cybernetics, Academy of Economic Studies Bucharest, Romania, e-mail: mihai.doinea@ie.ase.ro

For a positive variation of *x* factor, $S_{x,y}$ indicator varies from the starting point of $S_{x,y}$ with a positive value of $\frac{\delta}{y}$, the indicator showing its sensitivity characteristic.

From x < y, by adding a positive value to *x*, $\delta > 0$ and $x + \delta > y$, the new indicator is equal to:

$$S_{x+\delta,y} = \frac{min(x+\delta, y)}{max(x+\delta, y)} = \frac{y}{x+\delta} = \frac{x}{y} - \frac{x}{y} + \frac{y}{x+\delta} = S_{x,y} + \frac{y}{x+\delta} - \frac{x}{y}$$
$$= S_{x,y} + \frac{y^2 - x^2 - x\delta}{y(x+\delta)}$$

meaning that a variation of $\frac{y^2-x^2-x\delta}{y(x+\delta)}$ had produced compared with the $S_{x,y}$ initial indicator value.

For the positive variation of both factors, x and y, with δ, the indicator becomes:

$$S_{x+\delta,y+\delta} = \frac{\min(x+\delta, y+\delta)}{max(x+\delta, y+\delta)} = \frac{x+\delta}{y+\delta} = \frac{x}{y} + \frac{x+\delta}{y+\delta} - \frac{x}{y} = S_{x,y} + \frac{\delta(y-x)}{y(y+x)}$$

The value of the new indicator is greater than the initial one with a positive difference of $\frac{\delta(y-x)}{y(y+x)}$; the indicator being as well sensitive to the variation of both factors.

The property of **non-compensatory** of an indicator, it is present when at level variations of *x* and *y* factors, different variations of the indicator's value are produced.

If the variation of x and y factors are known, both variations expressed as follows:

$$x' = var_1 \cdot x;$$
$$y' = var_2 \cdot y;$$

and $x' < y'$ then the new value of the indicator is:

$$S_{x',y'} = \frac{\min(var_1 \cdot x, var_2 \cdot y)}{max(var_1 \cdot x, var_2 \cdot y)} = \frac{var_1 \cdot x}{var_2 \cdot y} = S_{x,y} \cdot \frac{var_1}{var_2}$$

If $var_1 = var_2$ then the indicator is compensatory, meaning that the influence of exogenous variables upon the internal factors doesn't influence the indicator measured value.
Non-catastrophic property lies in identifying situations when the indicator is not capable to reflect the values assimilated as input factors. In this case the indicator cannot evaluate some particular values of the exogenous *x* and *y* variables.

The following situations occur:

- *max(x,y)* = 0, meaning that both factors are 0; the value of the indicator is undetermined;
- *min(x,y) = 0*, case in which the indicator has the value $S_{x,y} = 0$, meaning that the planned or realized value for a factor was 0.

These situations determine some adjustments in terms of restrictions for the exogenous factors of the analyzed indicator.

The ***representativeness*** of this indicator is given by the appreciation, in time, of the concordance between the estimated level of the indicator and real level. If the indicator is used for the evaluation of the characteristics of a distributed application than is said to be representative if:

- for any level of exogenous variables, the indicator determines with accuracy the endogenous variable level;
- the indicator results are in concordance with the impact of measured characteristics of the distributed applications upon its users;
- based on the values of the endogenous variable measured by the indicator, a correct hierarchy of distributed applications can be created based on the analyzed characteristics.

In order to use this indicator to assess the effects of the security optimization process in distributed applications, calculated levels for the security application's quality are analyzed compared with the planned levels and the following situations are encountered:

- the estimated level is low and the application's security is affected by a multitude of existing threats that exploit vulnerabilities;
- the estimated level is lower but the methods and techniques implemented have a satisfactory security results in combating the threats, a new optimization stage being necessary;
- the estimated level is high but security requires continuous adjustment, the optimum values of the parameters not yet being achieved;
- the estimated level is high and the security does not require additional adjustments.

Situations in which the indicator determines a value for the level of security but the reality regarding the distributed application's security is different are excluded because the indicator is representative.

2. Quality analysis of distributed applications

3.

Quality analysis of distributed applications is made based on the refined metrics presented in [MDOIN11], of who's levels are determined and compared with the values targeted, in this way achieving a level of quality reported to the one initially aimed.

Usability describes the measure in which distributed applications are used by the users for achieving their specified goals in an efficient and satisfactory way, without any kind of impediment being given a standard context. It is the characteristic that describes the intuitiveness of the system so that the user not to need assistance when dealing with the operations [IVDO09]. Usability, *UM,* is described by the relation:

$$UM(\%) = \frac{1}{nop}\sum_{i=1}^{nop} q_i$$

where:

nop – total number of sessions registered in the system;

q_i – the degree realized by the user without assistance for completing the working session i.

The dimensional analysis of *UM* indicator is:

$$[UM] = \frac{[\sum_{i=1}^{nop} q_i]}{[nop]} = \frac{[q_i]}{[number\ of\ working\ sessions]}$$

Because of the fact that number of working sessions and q_i are dimensionless, it results the fact that *UM* is dimensionless. The possible values of *UM* indicator are situated in $UM\epsilon[0;1]$. If the working sessions done at the level of the system are made in full percentage by the users, without any king of assistance, $q_i = 1, \forall i = \overline{1, nop}$, then the maximum value of the indicator that describes a high level of usability is *UM=1*. In contrary, if all the sessions need total assistance, then *UM=0*. Based on table 2.1, with results referring to the decisional factor, the degree of achieving of the working session of the system without any assistance is the value of the *UM* indicator of 0.51, for *nop=129*.

Table 2.1 – The set of values for UM metric

No.	Degree of completeness for session *i* (q_i%)	No.	Degree of completeness for session *i* (q_i%)	No.	Degree of completeness for session *i* (q_i%)
1	0.8	**51**	1	**101**	0.8
2	0.2	**52**	0.8	**102**	0.6
…	…	**…**	…	**…**	…
49	0.6	**99**	0.8	**128**	0.8
50	0.8	**100**	0.8	**129**	0.8

Portability defines the degree in which, for achieving the portability process of a distributed application, the changing costs of the framework where it will be implemented and also those related to the changes made at the level of the existing functionalities, called integration costs, are justified related to the development costs of the application. The portability metric, *PM*, is determined based on the relation:

$$PM = 1 - \frac{C_{int}}{C_{dev}}$$

where:

C_{int} – integration costs resulted due to the process of application's portability;
C_{dev} – costs related to the development of the distributed application.

PM indicator is of type part from the whole and the dimensional analysis of it is:

$$[PM] = 1 - \frac{[C_{int}]}{[C_{dev}]} = 1 - \frac{[u.m.]}{[u.m.]} = 1$$

In the case of a positive scenario, if for the portability process of a distributed application, $C_{int} = 0$, then $PM = 1$, resulting a high level of portability of the application in the new environment. Otherwise, the values of PM indicator are distributed in the following manner:

$$PM = \begin{cases} 1, C_{int} = 0 \\ (0;1), 0 < C_{int} < C_{dev} \\ (-\infty; 0], C_{int} \geq C_{dev} \end{cases}$$

The scenario in which $PM \leq 0$ corresponds to the situation in which the integration costs are greater or equal to the development costs, representing an impractical solution, in this situation a newer system needs to be developed.

If $C_{int} = 3\ mld.$ și $C_{dev} = 34\ mld.$ then the value of the portability indicator *PM = 0.912*. The value of the indicator being close to 1, representing a high degree of portability, and the costs are justified.

Reliability in the effective way is defined as being the capacity of a distributed application of achieving the operations for which it was developed without any outliers' values, wrong at the level of the results, those errors representing the consequences of the interaction between the internal and external components. The reliability of a distributed application is defined by the *FM* indicator, type part from whole, it's value being dimensionless, using the formula:

$$FM = 1 - \frac{\sum_{k=1}^{ner} BN_k}{\sum_{i=1}^{nc}\sum_{j=1}^{nc} bt_{ij}}$$

where:

ner – number of errors found at the level of the application;
nc – number of components of the application;
bt_{ij} – number of bytes transferred between component *i* and *j*;
BN_k – number of bytes affected by error *k*.

The dimensional analysis of *FM* indicator is:

$$[FM] = 1 - \frac{[BN_k]}{[bt_{ij}]} = \frac{[bytes]}{[bytes]} = 1$$

The transfer matrix between the components of the distributed application, MCT, is defined as a matrix of which elements represents the number the bytes transferred between the components found at the lines and columns.

$$MCT = \begin{pmatrix} bt_{11} & bt_{12} & \dots & bt_{1j} & \dots & bt_{1\,nc} \\ bt_{21} & bt_{22} & \dots & bt_{2j} & \dots & bt_{2\,nc} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ bt_{i1} & bt_{i2} & \dots & bt_{ij} & \dots & bt_{i\,nc} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ bt_{nc\,1} & bt_{nc\,2} & \dots & bt_{nc\,j} & \dots & bt_{nc\,nc} \end{pmatrix}$$

The domain in which the reliability indicator takes values in $FM \in [0;1]$. When no errors are found in the application, *FM = 1*, in contrary, if at the level of the distributed application all transferred bytes are vitiated because of the errors, then *FM = 0*. Table 2.2 presents the number of bytes affected by the errors for a number of *ner = 1153* errors identified within the distributed application.

Table 2.2 – Values' set, FM metric

No. error	BN_k(KB)	No. error	BN_k (KB)
1	8	**501**	12
2	32	**502**	24
…	…	**…**	…
500	1	**1153**	19

The transfer matrix between the components of the distributed application, *MCT*, with *nc = 5* components, has the following values:

$$MCT = \begin{pmatrix} 0 & 46 & 0 & 0 & 0 \\ 0 & 32 & 160 & 228 & 240 \\ 0 & 28 & 56 & 0 & 0 \\ 0 & 32 & 0 & 64 & 286 \\ 0 & 0 & 0 & 120 & 64 \end{pmatrix}$$

The value of the indicator based on the values from the table 2.2 and the *MCT* matrix, having $\sum_{k=1}^{ner} BN_k = 522$ and $\sum_{i=1}^{nc} \sum_{j=i}^{nc} bt_{ij} = 1356$, is *FM = 0.62*.

Maintainability is the characteristic that offers the developers the ease in correcting the errors found after the process of deployment, adding with no great efforts new functionalities in the initial flow of the distributed application and modifying the existence ones. For approaching the maintainability of a distributed application two aspects are taken into consideration:

- the complexity of the source code written for the modules of the application for which are presented in the literature metrics like number of line codes, McCabe metric of complexity [IVDOL08] or Halstead complexity;
- the degree of modifications done at the source code based on which the maintainability indicator is weighted with the code's complexity.

Maintainability indicator, *MM*, reflects how much a distributed application is maintainable from the perspective of adding new functionalities, of modifying the existing ones for complying with the new requirements or even eliminating them, aggregated with the level of complexity defined by the graph associated to it. The changeability indicator, *CHM*, is given by the relation:

$$CHM = \frac{SL_0 - SL_R + SL_A}{SL_0 + SL_R + SL_A + SL_M}$$

where:

SL_0 – number of initial line codes;
SL_R – number of lines eliminated from the source code;
SL_A – number of lines added in the source code;
SL_M – number of lines modified.

Based on the dimensional analysis of *CHM* indicator, it results that the indicator is dimensionless:

$$[CHM] = \frac{[SL_0] - [SL_R] + [SL_A]}{[SL_0] + [SL_R] + [SL_A] + [SL_M]} = \frac{[linii\ cod]}{[linii\ cod]} = 1$$

The indicator for calculating the complexity is based on the number of ways between the starting node of the oriented graph associated to the distributed application to each functionality described as a node from this graph. The graph associated is represented in an interaction matrix *AM*. Each nod of the graph represents a functionality of the application, having *NF*, total number of functionalities.

$$AM = \begin{pmatrix} am_{11} & am_{12} & \dots & am_{1j} & \dots & am_{1\,NF} \\ am_{21} & am_{22} & \dots & am_{2j} & \dots & am_{2\,NF} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ am_{i1} & am_{i2} & \dots & am_{ij} & \dots & am_{i\,NF} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ am_{NF1} & am_{NF2} & \dots & am_{NF\,j} & \dots & am_{NF\,NF} \end{pmatrix}$$

The elements of the matrix $AM_{NF \times NF}$ are calculated using:

$$am_{ij} = \begin{cases} 1, \exists\ edge\ from\ i\ la\ j\ node \\ 0, \nexists\ edge\ from\ i\ la\ j\ node \end{cases}$$

Based on this matrix, the complexity indicator *GCM* of the graph, using the following formula:

$$GCM = \sum_{i=1}^{NF} \sum_{j=i}^{NF} am_{ij}$$

MM maintainability indicator is built based on *CHM* and *GCM* indicators, weighting the changeability with the complexity of the application's graph:

$$MM = \frac{1}{GCM} * CHM$$

The dimensional analysis of *MM* indicator is:

$$[MM] = \frac{[CHM]}{[GCM]} = 1$$

Because both *CHM* and *GCM* are dimensionless, it results the fact that *MM* indicator is also dimensionless. The two components from the indicator influence the results in the following way:

- if no changes are done within the system at the level of maintainability, then *CHM=1*, so the values of *CHM* decreases to 0 according to how much changes are realized and results that $CHM \in [0,1]$;
- the minimum value of the indicator for the complexity is *GCM = 1*, in the case when the distributed application has only one functionality, in contrary, as the number of components increases and the interactions among them also increase it results a level of complexity that increases without restrictions, $GCM \in [1; +\infty)$;
- if $gm = \frac{1}{GCM}, gm \in (0,1]$, represents the weight of the changeability indicator, then the maintainability indicator takes values in $MM \in (0; 1] \times [0; 1] = [0; 1]$.

Let SL_0 = 500, SL_R = 50, SL_A = 100 și SL_M = 150, then $CHM = 0.688$.
If the application has *NF = 5*, the functionalities implemented and *AM* matrix of the interaction between them is:

$$AM = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{pmatrix}$$

having the complexity *GCM = 12*, then *MM = 0.058*.
The quality level aggregated of the distributed application, *NCAD*, is analyzed with the use of the indicator presented in [IICM10], and described by the relation:

$$NCAD = \frac{\sum_{i=1}^{4} \frac{\min(NCC_{CAL_i}, NCC_{EXP_i})}{\max(NCC_{CAL_i}, NCC_{EXP_i})}}{\sum_{i=1}^{4} NCC_{EXP_i}}, 0 < NCC_{CAL_i} < NCC_{EXP_i}$$

Taking into consideration the results obtained by the distributed application, presented in table 2.3, the aggregated quality level has a value of *NCAD = 0.90.*

Table 2.3 – The quality level of the distributed application

Indicator	Measured value	Proposed value
UM	0.51	1
PM	0.912	1
FM	0.62	1
MM	0.058	0.08

The level of maintainability proposed is defined by the value of 0.08 because of the complexity level of the distributed application *GCM=12*, that limits obtaining the maximum value of the *MM* indicator, done only when *GCM=1*.

4. Optimization effects in distributed applications security

Starting from $S_{x,y}$ indicator, it correctly reflects the level measured if $x>0$ and $y>0$ restrictions are met. Using a weight system formed out of $psec_1$, $psec_2$, ..., $psec_{nca}$ that reflects the level of importance given to the characteristics, where *nca* represents the total number of characteristics taken into consideration, the aggregated indicator is given by the relation:

$$AGRS = \sum_{i=1}^{nca} psec_i * \frac{\min(QPLS_i, QRES_i)}{\max(QPLS_i, QRES_i)}$$

where:

$QPLS_i$ – the planned level for *i* characteristic;

$QRES_i$ – the achieved level for *i* characteristic;

$psec_i$ – the level of importance for *i* characteristic, having:

$$\sum_{i=1}^{nca} psec_i = 1$$

The dimensional analysis of *AGRS* indicator is:

$$[AGRS] = [importance\ level] \times \frac{[\text{achieved level}]}{[\text{planned level}]}$$

As level of importance is a dimensionless values and [achieved level]=[planned level], it results that [AGRS]=1.

A set of metrics defined for the evaluation of the security of distributed systems is developed in such a manner that the security aspects are followed step by step within the development cycle of the distributed systems.

These metrics allows the analyze of the security level and the effects of the security optimization process as a relative indicator of the value calculated at the initial time and the one measured when the process of optimization takes place.

The metrics of the characteristics that the values are measured are:

- the number of security problems appeared due to vulnerabilities at the level of analysis, ASI;

- the number of security problems appeared due to vulnerabilities at the level of development, DSI;
- the number of security problems appeared due to vulnerabilities at the level of testing, TSI;
- the number of security problems appeared due to vulnerabilities at the level of implementation, ISI;
- the number of security problems appeared due to vulnerabilities at the level of maintainability, MSI.

The levels of importance associated to the characteristics taken into account are calculated based on a study done on 40 specialists that give each analyzed characteristic marks with values between 1 and 5, where 1 represents a low impact, while 5 is a major impact, as it is presented in table 6.2.

Table 2.4 – The results of the survey upon security analysis

Questionnaire	**How much the vulnerabilities found at the levels of the development cycle of a distributed application have an impact upon the security level?**				
No. Stage	**Analysis**	**Development**	**Testing**	**Implementation**	**Maintenance**
1	5	5	4	5	4
2	4	3	2	4	3
3	5	3	5	2	5
4	3	1	1	4	4
…	…	…	…	…	…
40	4	5	3	5	3

Based on the information records from the specialists the values of the importance levels are calculated using the formula:

$$psec_j = \frac{\sum_{k=1}^{nr} a_{kj}}{\sum_{k=1}^{nr}\sum_{i=1}^{nca} a_{ki}},$$

where:

a_{ki} – the level of importance of characteristic *i* given by *k* user;
nr – the number of answers obtained;
nca – the number of characteristics.

The dimensional analysis for $psec_i$ is:

$$[psec_i] = \frac{[importance\ level]}{[importance\ level]} = 1$$

Table 6.3 presents the measured values of the characteristics' weights, of the planned values, the realized, according to the number of security problems due to the vulnerabilities of the stages of the development cycle of a distributed application, $QPLS_0$, $QRES_0$, and the values estimated for the indicators after the optimization.

Table 2.5 – The values of the indicators of the analyzed characteristics

	Planned $QPLS_{0/1}$	Measured $QRES_0$	Measured $QRES_1$	Weight $psec_i$
ASI	10	20	14	0.19
DSI	300	684	460	0.19
TSI	100	138	110	0.18
ISI	150	233	200	0.22
MSI	50	98	60	0.22

According to the values presented in table 6.3, the values of *AGRS* indicator, before and after the optimization stage, are:

- before the optimization:

$$AGRS_0 = 0.562$$

- after the optimization:

$$AGRS_1 = 0.771$$

After the optimization, the improvement of the security level is analyzed by calculating the indicator of measure of the security optimization's efficiency, *SOEM,* as:

$$SOEM_{1/0} = \frac{AGRS_1}{AGRS_0} * 100\% = 137{,}137\%$$

The dimensional analysis of $SOEM_{1/0}$ indicator is:

$$[SOEM_{1/0}] = \frac{[AGRS_1]}{[AGRS_0]}$$

As $[AGRS_1]$ and $[AGRS_0]$ are dimensionless values, it results that $[SOEM_{1/0}]$ =1.
The value of AGRS is improved with 37.137% comparing to its value calculated for the reference period of time.

5. Optimization as an automated process

In economy, [SULU02], reoptimization is presented as, giving a problem of linear programming of which optimum solution was calculated, it is desired to know that if, modifying the entry parameters, what can be reused from the initial solve of the problem. In this situation, reoptimization assumes following the steps:

- verifying the optimality of the current solution in the new conditions;
- determining the new solution in the case when the conditions of optimality are not met.

In informatics security, the process of reoptimization is seen as the moment of time in the optimization process, simultaneous with the moment of time when the security components fail under the level of security defined in the specifications. This drop under the quality level

is due to internal factors with a dynamic character within the distributed application or the demanding defined by the external environment.

The process of reoptimization is based on the following characteristics analyzed for establishing the moment of time when it is needed to be implemented:

- the quality planned level of the security component in the specifications;
- the quality measured level of the security component;
- the moment of time at which the measured level drops under a specified level.

For the component of access to resources, at the level of which an adaptive model of security optimization is implemented, a quality level is taken into consideration given by the percentage of correct classifications of the access to resources component within the system, *CCP*, defined by:

$$CCP = \frac{NCC}{NTC}$$

where:
NCC – the number of correct classifications;
NTC – total number of classifications.

The dimensional analysis of *CCP* indicator is:

$$[CCP] = \frac{[NCC]}{[NTC]} = \frac{[number\ of\ classifications]}{[number\ of\ classifications]} = 1$$

The model uses a neuronal algorithm based on a perceptron of which weights are calculated in the training process.

Let t_0 be the moment represented by the time when the training process of the perceptron takes place so that CCP_0 the percentage of correct classifications to be 95%.

$$t_0: CCP_0 \geq 95\%$$

In the case when, due to an unpredictable and heterogeneous character of the users, this percentage of correct classifications drops under the level of CCP_0, a reoptimization is done at the way the access to resources is done so that the weights of the neuronal system to be recalculated. This process is based on further information found in the data base at the moment of t_1 that helps to a new configuration of the neuronal network, adjusting the weights so that CCP_1 to verify the relation $CCP_1 > CCP_0$.

Reoptimization is seen as an automated process so that each time the quality level represented by *CCP* indicator drops under the defined threshold, the network is retrained, forming its new values that leads to a superior percentage of classification.

Let $f_i: N \to [0; 1]$ be the function assigned to the i neuronal network defined on the set of natural numbers, represented by the variables time expressed in number of days, taking values in $[0; 1]$, represented by the percentage of correct classification done by the i neuronal network, with:

$$f_i(u) = \frac{NCC_i(u)}{NTC_i(u)}$$

where:

$NCC_i(u)$ – number of correct classifications using the *i* neuronal network at the moment of time *u*;

$NTC_i(u)$ – total number of classifications done by the *i* neuronal network at the moment of time *u*.

If at the moment of t_0 the *k* neuronal network implemented is generating a number of correct classifications determined by the function $f_k(t_0) = 0.95$, the reoptimization moment t_1 is the one where $f_k(t_0) > f_k(t_1)$.

In this situation, after the reoptimization, while adjusting the weights of the *k* neuronal network, the *k+1* neuronal network is created of which weights leads to the verification of the following $f_k(t_0) \leq f_{k+1}(t_1)$.

This process is presented in figure 3.

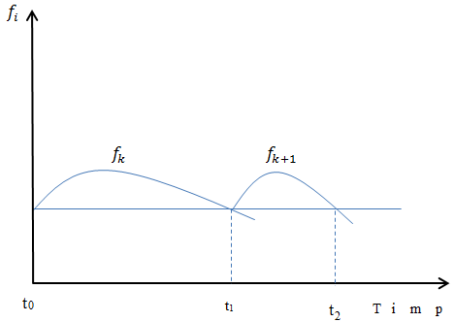

Figure 3 – Determining the reoptimization moment

The reoptimization, seen as an automated process, is redone at the moment t_2, point in which $f_k(t_0) > f_{k+1}(t_2)$. The process of reoptimization within the component of access to resources has the role of using the existence resources found in the initial optimum solution and, based on the modifications appeared at the entry parameters, a new network is created having a new set of weights that achieves a classification according to the values presented in the specifications.

In the distributed application, is developed the component that realizes the reoptimization at the level of the neuronal network based on the information found in the system at the moment of time when $f_k(t_0) > f_k(t_1)$, adjusting the system's weights so that $t_1 : CCP_1 \geq 95\%$.

6. Conclusions

Optimization is an important process in the development and distributed applications' evolution. For the optimization process ends up with maximum efficiency another parallel process must record simultaneously the evolution of the first one to see whether if, based on a set of metrics, the trajectory given by the optimization is leading to better results that can justify the resources involved along the process.

Bibliography

[IICM10**]** I. Ivan, C. Ciurea, D. Milodin – *Collaborative Educational System Analysis and Assessment*, Proceedings of The Third International Conferences on Advances in Computer-Human Interactions, ACHI 2010, February 10-16, 2010, Saint Maarten, Netherlands Antilles, ISBN 978-0-7695-3957-7

[MDOIN11] M. Doinea – *Optimizarea securității aplicațiilor informatice distribuite*, Academia de Studii Economice, Teza de doctorat, Septembrie, 2011, București, Romania

[IVDO09] I. Ivan, M. Doinea – *Social Networks Security*, The Journal of Applied Collaborative Systems, Vol. 1, No. 2, 2009, pp. 101 – 110, ISSN 2066-7450

[IVDOL08] I. Ivan, M. Doinea, L. Săcuiu – *Evaluating security of non-homogenous distributed applications*, 4th International Conference on Applied Statistics, Bucharest, November 2008, ISBN 1018-046x

[SULU02] C. Ratiu Suciu, F. Luban – *Modelarea si simularea proceselor economice – sinteza*, Editura ASE, Bucuresti, 222 pag., 2002, ISBN 9735941864